IN THE CLAIMS:

Please **AMEND** the claims as follows

1.   (Currently Amended) A method of executing a ~~method~~ set of source code instructions to enable memory associated with objects not referenced external to the executed ~~method~~ set of source code instructions to be reclaimed upon completion of execution of the executed ~~method~~ set of source code instructions, comprising:

obtaining a data structure ~~including~~ storing one or more addresses of ~~source code~~ one or more of the set of source code instructions ~~method~~ that creates one or more local objects, the one or more local objects being created during execution of the set of source code instructions ~~method~~ and are not referenced outside the set of source code instructions ~~method~~;

obtaining a next source code instruction from ~~the source code of~~ the set of source code instructions ~~method~~;

determining whether an address of the next source code instruction is in the data structure;

when the address of the next source code instruction is in the data structure including one or more addresses of ~~source code of the method~~ one or more of the set of source code instructions that creates one or more local objects upon execution of the one or more of the set of source code instructions, creating a local object on a local heap of memory using the next source code instruction associated with the address of the next source code instruction such that local objects are stored in memory separately from non-local objects; and

when the address of the next source code is not in the data structure including one or more addresses of ~~source code of the method~~ one or more of the set of source code instructions that creates one or more local objects, creating a non-local object on a non-local heap of memory using the next source code associated with the address of the next source code such that non-local objects are stored in memory separately from local objects.

2.   (Currently Amended) The method as recited in claim 1, wherein determining whether the address of the next source code instruction is in the data structure comprises:

determining whether a program counter of the next source code is in the data structure.

3. (Currently Amended) The method as recited in claim 1, further comprising:

    reclaiming memory associated with the local heap upon termination of execution of the ~~method~~ set of source code instructions.

4. (Currently Amended) The method as recited in claim 1, further comprising:

    returning memory associated with the local heap to a pool of available memory upon termination of execution of the set of source code instructions ~~method~~.

5. (Currently Amended) The method as recited in claim 1, further comprising:

    creating a dynamic structure adapted for storing dynamic information related to ~~method~~ execution of the set of source code instructions; and

    associating the local heap with the dynamic structure.

6. (Original)   The method as recited in claim 5, wherein associating the local heap with the dynamic structure comprises extending a pointer from the dynamic structure to the local heap.

7. (Original)   The method as recited in claim 5, wherein the dynamic structure is a Java frame.

8. (Original)   The method as recited in claim 7, wherein the Java frame is a data structure in a Java interpreter.

9. (Currently Amended) The method as recited in claim 8, further comprising:

    removing the Java frame from memory when execution of the ~~method~~ set of source code instructions terminates.

10. (Original)   The method as recited in claim 1, further comprising:

    allocating a free chunk of available memory as the local heap for storage of one or more local objects.

11.   (Currently Amended) The method as recited in claim 1, wherein the local heap comprises one or more chunks of memory, wherein creating a local object on a local heap of memory using the next source code instruction comprises:

determining whether the local heap contains available memory for storage of the local object;

when the local heap contains available memory sufficient for storage of the local object, creating the local object in one of the chunks of memory;

when the local heap does not contain available memory sufficient for storage of the local object, allocating a new chunk, associating the new chunk with the local heap, and storing the local object in the new chunk.

12.   (Original)   The method as recited in claim 11, wherein associating the new chunk with the local heap comprises providing a pointer to the new chunk such that the local heap is composed of a linked list of memory chunks.

13.   (Previously Amended)   The method as recited in claim 1, wherein obtaining a data structure including one or more addresses of source code that creates local objects comprises:

obtaining an attribute_info structure from a Java class file, wherein the data structure is the attribute_info structure of the Java class file.

14.   (Currently Amended) The method as recited in claim 1, wherein the set of source code instructions comprises bytecodes.

15.   (Original)   The method as recited in claim 8, wherein the bytecodes are Java bytecodes.

16.   (Currently Amended) A computer-readable medium for executing a ~~method~~ set of source code instructions to enable memory associated with objects not referenced external to the executed ~~method~~ set of source code instructions to be reclaimed upon completion of execution of the executed ~~method~~ set of source code instructions, comprising:

instructions for obtaining a data structure ~~including~~ storing one or more addresses of

~~source code~~ <u>one or more</u> of the <u>set of source code instructions</u> ~~method~~ that creates one or more local objects, the one or more local objects being created during execution of the <u>set of source code instructions</u> ~~method~~ and are not referenced outside the <u>set of source code instructions</u> ~~method~~;

instructions for obtaining <u>a</u> next source code <u>instruction</u> from ~~the source code of~~ the <u>set of source code instructions</u> ~~method~~;

instructions for determining whether an address of the next source code <u>instruction</u> is in the data structure;

instructions for when the address of the next source code <u>instruction</u> is in the data structure including one or more addresses of ~~source code of the method~~ <u>one or more of the set of source code instructions</u> that creates one or more local objects <u>upon execution of the one or more of the set of source code instructions</u>, creating a local object on a local heap of memory using the next source code <u>instruction</u> associated with the address of the next source code <u>instruction</u> such that local objects are stored in memory separately from non-local objects; and

instructions for when the address of the next source code is not in the data structure including one or more addresses of ~~source code of the method~~ <u>one or more of the set of</u> <u>source code instructions</u> that creates one or more local objects, creating a non-local object on a non-local heap of memory using the next source code associated with the address of the next source code such that non-local objects are stored in memory separately from local objects.

17. (Currently Amended) An apparatus for executing a ~~method~~ <u>set of source code instructions</u> to enable memory associated with objects not referenced external to the executed ~~method~~ <u>set of source code instructions</u> to be reclaimed upon completion of execution of the executed ~~method~~ <u>set of source code instructions</u>, comprising:

means for obtaining a data structure ~~including~~ <u>storing</u> one or more addresses of ~~source code~~ <u>one or more</u> of the <u>set of source code instructions</u> ~~method~~ that creates one or more local objects, the one or more local objects being created during execution of the <u>set of source code instructions</u> ~~method~~ and are not referenced outside the <u>set of source code instructions</u> ~~method~~;

means for obtaining a next source code instruction from ~~the source code of~~ the set of source code instructions ~~method~~;

means for determining whether an address of the next source code instruction is in the data structure;

means for when the address of the next source code instruction is in the data structure including one or more addresses of ~~source code of the method~~ one or more of the set of source code instructions that creates one or more local objects upon execution of the one or more of the set of source code instructions, creating a local object on a local heap of memory using the next source code instruction associated with the address of the next source code instruction such that local objects are stored in memory separately from non-local objects; and

means for when the address of the next source code is not in the data structure including one or more addresses of ~~source code of the method~~ one or more of the set of source code instructions that creates one or more local objects, creating a non-local object on a non-local heap of memory using the next source code associated with the address of the next source code such that non-local objects are stored in memory separately from local objects.

18.   (Currently Amended) An apparatus for executing a ~~method~~ set of source code instructions to enable memory associated with objects not referenced external to the executed ~~method~~ set of source code instructions to be reclaimed upon completion of execution of the executed ~~method~~ set of source code instructions, comprising:

a processor; and

a memory, at least one of the processor and the memory being adapted for:

obtaining a data structure ~~including~~ storing one or more addresses of ~~source code~~ one or more of the set of source code instructions ~~method~~ that creates one or more local objects, the one or more local objects being created during execution of the set of source code instructions ~~method~~ and are not referenced outside the set of source code instructions ~~method~~;

obtaining a next source code instruction from ~~the source code of~~ the set of source code instructions ~~method~~;

determining whether an address of the next source code <u>instruction</u> is in the data structure;

when the address of the next source code <u>instruction</u> is in the data structure including one or more addresses of ~~source code of the method~~ <u>one or more of the set of source code instructions</u> that creates one or more local objects <u>upon execution of the one or more of the set of source code instructions</u>, creating a local object on a local heap of memory using the next source code <u>instruction</u> associated with the address of the next source code <u>instruction</u> such that local objects are stored in memory separately from non-local objects; and

when the address of the next source code is not in the data structure including one or more addresses of ~~source code of the method~~ <u>one or more of the set of source code instructions</u> that creates one or more local objects, creating a non-local object on a non-local heap of memory using the next source code associated with the address of the next source code such that non-local objects are stored in memory separately from local objects.

19.   (Currently Amended) The method as recited in claim 1, further comprising:

compiling the ~~method~~ <u>set of source code instructions</u> to generate the data structure.

20.   (Currently Amended) The method as recited in claim 19, wherein the source code is generated when ~~the~~ <u>a</u> method is compiled.

21.   (Previously Added)   The method as recited in claim 20, wherein the source code comprises bytecodes.

22.   (Previously Added)   The method as recited in claim 3, wherein reclaiming memory is performed during garbage collection.

23.   (Previously Added)   The method as recited in claim 22, wherein the garbage collection is mark and sweep garbage collection.

24.   (Previously Amended)   The method as recited in claim 13, further comprising:

performing class file generation such that information from a second data structure is
stored in the attribute_info structure of the Java class file.

25.   (Previously Added)   The method as recited in claim 13, further comprising:
        performing class file generation such that the data structure is generated.

26.   (Cancelled)

27.   (Previously Added)   The method as recited in claim 1, further comprising:
        generating the data structure.

28.   (Previously Added)   The method as recited in claim 27, further comprising:
        performing live-dead analysis on the source code to identify a set of dead objects,
each of the set of dead objects being a local object.

29.   (Currently Amended) The method as recited in claim 1, further comprising:
        executing or interpreting the next source code instruction.

Please ADD claims as follows:

30.   (Currently Added)   The method as recited in claim 1, wherein the data structure is
generated during compilation of the set of source code instructions.